REMARKS

Claims 1-22 remain pending in the present application. Reconsideration and allowance of the application and presently pending claims are respectfully requested.

I.   Response to 35 U.S.C. §102 Rejection

Claims 1, 6-11, 14, and 21 stand rejected under 35 U.S.C. §102(b) as allegedly being anticipated by *Chakradhar et al.* (U.S. Patent No. 5,726,996). Applicants respectfully traverse this rejection.

A.   *Chakradhar et al.*

Chakradhar et al. discloses a process for test sequence compaction and test cycle reduction. The process identifies "bottlenecks" that prevent test vector compaction and test cycle reduction. Subsequent test sequences are generated with the aim to eliminate the bottlenecks of the initially generated test sequences. A sliding anchor frame technique can be used to extend partially specified test sequences for detecting additional faults. (See the abstract.)

In an effort to compact test sequences and reduce the number of test cycles, *Chakradhar et al.* uses a "subset-selection approach," which involves selecting a subset of the test sequences from a given set that detects all target faults (col. 4, lines 35-39). Examples of the subset-selection approach include a "set-covering approach" and an "extended set-covering approach." The set-covering approach can specify unspecified values by merging multiple test sequences into a single test sequence. The extended set-covering approach arbitrarily specifies values for unspecified inputs (col. 4, lines 47-58).

As is known, if test vectors are partially specified, it is possible to merge two test vectors into a single vector if the corresponding primary inputs of the two vectors do not have conflicting value assignments. If the test vectors cannot be merged, then the size of the test set cannot be further reduced using the set-covering approach (col. 5, lines 13-19). These observations lead to *Chakradhar's* definition of vector compaction bottlenecks CB1 and CB2 (col. 5, lines 22-25). Similar observations lead to sequence compaction bottlenecks SCM1 and SCM2 (col. 5, lines 49-53).

To eliminate bottlenecks of a test vector, *Chakradhar et al.* appear to use fault simulation and attempt to extend the test vector to detect a bottleneck fault. When the

2

bottleneck has been eliminated, the test vector can be dropped. See Example 1 and Example 2 (col. 5, line 63 through col. 6, lines 58).

Starting in col. 6, line 60, *Chakradhar et al.* describes "test cycle bottlenecks," which can be identified as well. *Chakradhar et al.* teaches that test sets having the same number of test vectors can require vastly different numbers of test application cycles. For this reason, optimizing a test set for size is different from optimizing for the number of test cycles (col. 9, lines 4-5). The smallest test set may not always require the least number of test cycles (col. 9, lines 47-48).

In Fig. 11, *Chakradhar et al.* shows pseudo-code of a bottleneck elimination process framework. The bottleneck framework is performed while undetected faults still exist. In this process, an undetected fault is selected and a test sequence is generated for this fault (col. 10, lines 7-10). The test sequence is extended by suitably specifying unspecified signal values in the test sequence in order to detect additional faults (col. 10, lines 33-35). Next, a bottleneck elimination procedure is performed to further extend the test sequence for eliminating essential faults that are bottlenecks of prior test sequences (col. 10, lines 54-57). Fig. 11 further includes fault simulation, dropping, and trimming procedures. If every prior test sequence has bottlenecks that could not be eliminated by the current test sequence, then a procedure is run to determine the bottlenecks for the current test sequence (col. 11, lines 30-33).

B.   Claim 1

Independent claim 1 includes generating at least one test vector that defines values for detecting at least one target fault, the values comprising only a portion of the bits of the at least one test vector. A remainder of the bits in the at least one test vector are unspecified bit positions. Claim 1 further includes *"setting the values of a plurality of the unspecified bit positions using a non-random filling methodology."* The Office Action alleges that *Chakradhar et al.* sets unspecified bit positions using a non-random filling methodology. Applicants respectfully traverse this allegation.

It should be noted, first of all, that the term "filling" in the art of automatic test pattern generation (ATPG) has a particular meaning that is known to those skilled in the art. Filling is not the same thing as "assigning," "specifying," or "generating" values for bits that were previously unspecified. To define these different terms, the following is given as background.

3

During test sequence generation, values 0 or 1 are assigned to specific bit positions in order to detect faults. This group of values for detecting a particular target fault is commonly referred to as a "test cube." The test cube, generated by a test generator, includes both input values for applying value to create a particular fault condition and output values for observing whether the proper response to the fault condition has been met. Since a test cube may only test one fault condition and there are sometimes thousands of fault conditions to test, the test cube by itself is not an efficient test vector. Thus, it has been known to merge or compact multiple test vectors together to form a single test vector, thereby producing a more efficient test set. Other techniques, such as test vector extending, fault simulating, and trimming, can be used to compact the test set to an optimal size.

Mainly, there are three main goals in the context of ATPG:

1) maximize fault coverage;

2) minimize the number of test vectors; and

3) minimize the test cycle time.

Chakradhar et al. identifies these goals in an attempt to create a test set involving fewer test vectors and/or fewer test cycles in order to increase the rate of testing.

It is also known in the art of ATPG that once every reasonable effort has been made to compact the test vectors as efficiently as possible, there will still remain some bits that are unspecified. The conventional strategy at this point is to randomly "fill" those bits. The present application describes that filling can be done before, during, or after compaction and is even independent of compaction (p. 23, lines 6-9). Conventional filling methods typically involve randomly setting values for the don't care unspecified bits. By randomly filling, certain advantages can be attained. For example, random filling may provide a greater fault coverage (p. 24, line 4-5). However, as described in the present application, the inventors have discovered an alternative to "random filling" which has not been attempted in the prior art. Although non-random filling will not necessarily help meet the three goals mentioned above, and can even work against these goals, non-random filling provides other unique advantages. By "non-random filling," test sets can be more easily "compressed," thereby requiring less memory for storage (p. 24, lines 6-8).

At this point, it should also be noted that "compaction" and "compression" are two different concepts altogether. In ATPG, compaction is the process of merging or

combining two or more test vectors in a single test vector. Compression, however, is a process for actually shrinking the size or length of a test vector. In compression, a test vector having a certain length, depending, for instance, on the number of scan FFs in a scan chain, can be compressed to a shorter length. A compressed test vector can be more easily stored into memory and even can be delivered to the ATE more easily. In this case, the compressed test set being delivered to the ATE can therefore be considered as being outside the context of ATPG. Of course, a compressed test vector will normally have to be uncompressed during the actual testing of the DUT, which may be done by the ATE. It is believed that the concept of compressing a test vector, as described in the present application, does not appear in the prior art.

Referring again to claim 1 of the present application, the Office Action points to col. 10, lines 14-16 of *Chakradhar et al.*, which states: "We assume that the test generator does not randomly assign values to primary inputs and scan FFs that were left unspecified in the test sequence." From this sentence, the Office Action seems to conclude that if the test generator does not randomly assign values that were left unspecified, then a deterministic method necessarily is used in the assignment of the unspecified values and the deterministic method for assigning values is a non-random assigning method. Applicants wish to again clarify that claim 1 includes using a non-random filling methodology, which in ATPG is not the same as assigning values during normal test generation.

Also, this sentence in *Chakradhar et al.* is used in a paragraph that is describing the use a test generator for generating a test sequence for a selected undetected fault (col. 10, lines 7-10). Of course, a test sequence for detecting one fault will eventually be extended to detect additional faults. Therefore, to make it easier to detect additional faults, *Chakradhar et al.* does not randomly assign values for the unspecified values nor would there be a reasonable explanation for doing so since specifying unspecified bits at this point will only impede compaction. The test sequence is instead fault simulated to identify other fortuitously detected faults (col. 10, lines 21-22). With respect to ATPG, *Chakradhar et al.* does not describe "filling" as defined in the claims, but instead describes specifying values for the bits in order to detect undetected faults.

Furthermore, the Examiner's conclusion is actually contradictory to the purposes of *Chakradhar et al.* In col. 10, lines 11-20, *Chakradhar et al.* teaches that

not all primary inputs and scan FFs have to be assigned values 0 or 1 to detect the target fault and as few primary inputs and scan FFs are assigned values as possible to detect that target fault. Therefore, with many bits left unspecified, it will be easier to fault simulate the test sequence to detect additional faults and will also be easier to compact the test sequence with other sequences. If unspecified values were filled, then *Chakradhar*'s goal of test sequence compaction would be impossible to achieve at this early stage.

The Office Action further include a reference to col. 13, lines 65-67 of *Chakradhar et al.* This passage states that "[a] popular technique is to randomly or judiciously specify unspecified signals ... in order to drop test vectors or sequences from the test set." The Examiner appears to interpret "judiciously" as non-random. However, this again is referring to a different concept in the art of ATPG, in which "judiciously specifying unspecified signals" merely refers to conventional test generation and compaction procedures, but does not refer to <u>filling</u>.

Referring again to the passage in *Chakradhar et al.*, col. 10, lines 14-16, the Office Action seems to imply that "assigning" is equivalent to "filling." However, it is common knowledge to one of skill in the art of ATPG that filling has a particular meaning that is distinct from specifying or assigning values during generation or compaction. The Applicants therefore traverse the implication that assigning is the same as filling. Furthermore, the Office Action defines "does not randomly assign" to have the same meaning as the claimed element "non-random filling." The Applicants disagree with this interpretation since *Chakradhar et al.* evidently does not randomly assign value because it would make fault simulation and test sequence extension impossible. In contrast to the interpretation in the Office Action, Applicants refer to *Chakradhar et al.*, col. 10, lines 15-18, where the test generator does not randomly assigned values, and even "should assign values to as few primary input signals and scan FFs as possible to detect the target fault." In this context, *Chakradhar et al.* is still referring to generating a test sequence to detect <u>a single selected target fault</u>.

The Examiner is reminded that each reference must be considered as a whole and should not be interpreted apart from the rest of the specification. In this regard, it should be noted that *Chakradhar et al.* actually teaches a process for test sequence compaction and test cycle reduction, and further identifies bottlenecks that prevent test vector compaction and test cycle reduction. Subsequent test sequences are

6

generated for eliminating the bottlenecks of the "initially generated test sequences." *Chakradhar et al.* is silent concerning the aspect of "filling" unspecified bits that are left unspecified after the test vectors have been compacted. As is known, conventional ATPG techniques would merely randomly fill the unspecified bits at this point in the test generation process. *Chakradhar et al.* thus fails to address the aspect of claim 1 of setting values using a "non-random filling methodology" as claimed.

In addition, the passage in *Chakradhar et al.* describing "[a] popular technique is to randomly or judiciously specify unspecified signals..." appears to refer to conventional test generating and processing. The statement describes a <u>popular technique</u>, implying that it should be easily found in the prior art. If the Examiner's interpretation is correct, then there should be plenty of evidence in the prior art that supports "non-random filling." However, what is actually found in the prior art is <u>conventional test generation and related processing</u> such as compaction and fault simulation. For this reason, Applicants believe that this passage fails to teach "non-random filling" as defined in the claims of the present application.

The Examiner states that if the test generator does not randomly assign values that were left unspecified, then a "deterministic method necessarily is used" in the assignment of values that were left unspecified. Actually, *Chakradhar et al.* teaches just the opposite in col. 13, lines 22-24, in which the compaction techniques do not attempt to specify values to unspecified signals in the test sequences in order to merge test sequences. *Chakradhar et al.* does not necessarily have to use an assigning or deterministic method for these unspecified values in this case. Instead, they are left as unspecified signals so that test sequences can be merged. Nevertheless, this again is not referring to "filling" as claimed and is therefore irrelevant to claim 1.

The Examiner seems to assume that "judiciously" specifying unspecified signals is the same as Applicants' step of setting unspecified bit positions "using a non-random filling methodology." Actually, *Chakradhar et al.* refers to these techniques as examples of extended set-covering approaches" (col. 14, lines 2-3). *Chakradhar et al.* also defines these extended set-covering approaches in col. 4, lines 51-58, where it states that a "subset-selection approach that is allowed to change the test sequences by arbitrarily specifying a 0 or 1 value for each unspecified input is referred to as an extended set-covering approach (for example, an extended set-covering approach could use a test generator to judiciously specify unspecified

values..." Therefore, judiciously specifying according to *Chakradhar et al.* is actually done by "arbitrarily specifying a 0 or 1 value." This process of arbitrarily specifying values appears to be consistent with other conventional methods for extending a test sequence for greater fault coverage. This process of arbitrarily specifying is not the same as the claimed feature of setting unspecified bit positions "using a non-random filling methodology."

For at least the reason that *Chakradhar et al.* fails to disclose "*setting the values of a plurality of the unspecified bit positions using a non-random filling methodology,*" Applicants contend that this reference does not anticipate the present claims. Anticipation requires that each and every element of the claimed invention be disclosed in a single prior art reference. *See e.g., In re Paulsen,* 30 F.3d 1475, 31 USPQ 2d 1671 (Fed. Cir. 1994); *In re Spada,* 911 F.2d 705, 15 USPQ 2d 1655 (Fed. Cir. 1990).

Dependent claims 6-11, 14, and 21 are believed to be allowable for at least the reason that these claims depend from allowable independent claim 1. *In re Fine,* 837 F.2d 1071, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). Furthermore, claim 7 is directed to setting a second plurality of unspecified bit positions using a random filling methodology. In this case, both random and non-random filling methodologies are used to fill different fractions of the unspecified bit positions. This aspect is not disclosed in *Chakradhar et al.*

II.   Response to 35 U.S.C. §103 Rejection

Claims 2-5, 12, 13, 15-20, and 22 stand rejected under 35 U.S.C. §103 as allegedly being unpatentable over *Chakradhar et al.* (U.S. Patent No. 5,726,996). Applicants respectfully traverse this rejection because *Chakradhar et al.* does not teach or suggest all of the claim limitations of independent claims 1 and 15. Furthermore, there is no suggestion or motivation in the prior art for modifying *Chakradhar et al.* to include the lacking features. Also, modifying the reference in such a way would destroy the intended functions taught in *Chakradhar et al.,* thereby providing no reasonable expectation of success.

In order to make a proper *prima facie* case of obviousness, three basic criteria must be met, as set forth in MPEP 706.02(j). First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available

to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art references, when combined, must teach or suggest all the claim limitations. The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on Applicant's disclosure.

As mentioned above, *Chakradhar et al.* fails to teach or suggest *"setting the values of a plurality of the unspecified bit positions using a non-random filling methodology"* as claimed in independent claim 1. There is also no motivation to modify this reference to include this claimed feature. *Chakradhar et al.* is silent concerning the concept of "filling" and does not provide any evidence that might modify any convention random filling methodology to lead one of ordinary skill in the art to the claimed non-random filling methodology. With respect to the concept of assigning or specifying values, which as described above is distinct from filling, *Chakradhar et al.* appears to avoid and even undo any type of assigning of unnecessarily specified values. Any modification in this regard would destroy the concepts of compaction taught by *Chakradhar et al.* and would not provide a "reasonable expectation of success" in accordance with MPEP 706.02(j).
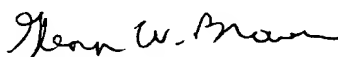
Similarly, *Chakradhar et al.* fails to teach or suggest *"second means for setting a plurality of the values of the unspecified bit positions using a non-random filling methodology"* as claimed in independent claim 15. *Chakradhar et al.* fails to suggest any structure or means that sets unspecified bit positions using a <u>non-random filling</u> methodology. *Chakradhar et al.* is actually silent concerning any type of filling methodologies. One would imply therefore that, at most, *Chakradhar et al.* might use means for setting unspecified values using a conventional random filling methodology. There is no evidence in the prior art that would suggest any modification from this convention technique. Therefore, modification of a random filling technique to a non-random filling technique, as taught in the present application, could only be realized using impermissible hindsight of Applicant's own invention.

CONCLUSION

In light of the foregoing amendments and for at least the reasons set forth above, Applicant respectfully submits that all rejections have been traversed, and that the pending claims 1-22 are in condition for allowance. Favorable reconsideration and allowance of the present application and all pending claims are hereby courteously requested. If, in the opinion of the Examiner, a telephonic conference would expedite the examination of this matter, the Examiner is invited to call the undersigned at (770) 933-9500.

Respectfully submitted,

Glenn W. Brown
Reg. No. 51,310

THOMAS, KAYDEN,
HORSTEMEYER & RISLEY, L.L.P.
Suite 1750
100 Galleria Parkway N.W.
Atlanta, Georgia 30339
(770) 933-9500

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail, postage prepaid, in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on .

Signature – Evelyn Sanders

10